Day 4 Exercises

Section G. Examples that require the use of Arrays

Note: I've just added question 4 onwards which you may want to start first as the problem are simpler. I keep the original numbering so that people who did the problem before doesn't have to change the numbering.

1. A company records its monthly sales information in an array of size 12; where Sales[0] represents January sales, Sales[1] is February sales etc. After entering the data the company wants to perform some queries on the data. Write a program that would do the following:
   a. Take in the sales for the 12 months.
      Note: You may use the array initialisation inside the program for storing these.
   b. Print the month when Maximum Sales is recorded.
      Note: You may just print 0, 1 etc. for Jan Feb etc…
   c. Print the month where Minimum Sales is recorded.
   d. Print the average monthly sales for the year.

2. Write a C# program that would sort a numeric array in descending order using the simplified selection sort method described.

   At the end of each pass print out the array to know the progress!

   Example for Multidimensional Array

3. The marks of students are stored in a two dimensional array with the subjects represented in columns and the students in the rows. That is Row 1 would pertain to Student 1 and the scores that this student has obtained is stored in various columns in row 1. Assuming that there are 12 students in a class and 4 subjects, write a program that would do the following:
   a. Compute the total marks obtained each student.
   b. Compute the class average (and standard deviation* - optional) of Marks for each subject.
   c. Determine the overall average of marks for the whole class for each subjects.
   d. Optional – make sure that your code works regardless of the number of students and the number of subjects. This means that you cannot make any assumption in the code that there are 12 students and 4 subjects.

   Note:

   * calculation of standard deviation is not required for first time exercise you may only compute the average – those needing additional practice may compute standard deviation)

Standard Deviation is square root of variance where Variance is given by:
VARIANCE = { [SUM OF $(X_i - M)^2$] / N }; i = 1 to N
N is number of data elements (X_i) and
M is mean (average).

Subject 1	Subject 2	Subject 3	Subject 4	Total	Avg
56	84	68	29	237	59.25
94	73	31	96	294	73.5
41	63	36	90	230	57.5
99	9	18	17	143	35.75
62	3	65	75	205	51.25
40	96	53	23	212	53
81	15	27	30	153	38.25
21	70	100	22	213	53.25
88	50	13	12	163	40.75
48	54	52	78	232	58
64	71	67	25	227	56.75
16	93	46	72	227	56.75

Average per subject:

59.16667	56.75	48	47.41667

New Questions

4. Create an array of a certain size and fill it up with numbers. You can decide on the size and the numbers can be either generated randomly or manually or programmatically following a pattern.
   After creating this array, print the content of the array to the screen in the following format:

```
[ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ]
```

5. You can reuse the array from question 4. Ask user for a number and return the position of that number in that array. Return -1 if the number cannot be found.

```
Enter a number: 5
Number 5 is found in the array at the element no 5.
```

   For the exercise you should write your own implementation. However, there's a method Array.IndexOf() which does something similar.

6. Create and fill up an array with number and then write a code to sort the array in ascending or descending order.

   For the exercise you should write your own implementation. However, there's a method Array.Sort() which does something similar.

7. Create two sets of array with the same size, let's call the array **name** and **score**. Let the first array contains player name and the second array contains their score. We assume that score[0] is the score of the player in name[0].

   Your task is to print the players with their scores with the player names sorted in ascending order.

```
Name       Score
Alice      100
Bob        90
Charlie    120
Dennis      80
Eli         76
Frank       66
Gina       88
```

8. Searching problem in multi-dimensional array.
   Create a two-dimensional array and fill it up with number. Ask user for a number and report back with the index for that number.

```

Enter a number: 5
Number 5 is found in the array at [4, 2].

```

<u>Section H. Examples involving Modularization using Static Methods</u>

Note: *For the following exercises it is possible that an equivalent method is available in the .NET api. However, as an exercise you are required to procedurally code these methods for practice and submission.*

1. Write a static method ReadInteger(string message) that would return an integer. The method should prompt the user with the message, get the input from the user using Console.ReadLine and if the input can be parsed into integer return the integer. If the input cannot be parsed into integer, the method should repeat the prompt until the user enter an integer input.

2. Write a static method PrintArray(int[] arr) that doesn't return any value. This method should print out all the elements of an array to the console in any format that you prefer.

3. Write a static method: InString(string s1, string s2) that would return a boolean. The method should find if the string s2 occurs in s1 and return true if it occurs else it would

return false. After you write the InString method, write a Main method that calls InString method.

S1	S2	Output
The brown fox	O	True
The brown fox	FOX	True
The brown fox	bRO	True
T	bRO	False
The bras basah complex	bRO	False

4. Write a static method: FindWord(string s1, string s2) that would return an integer. The method should find if the string s2 occurs in s1 and return an integer that would indicate the starting position of the word s2 in s1. If the word does not occur the return value should be –1.

S1	S2	Output
The brown fox	o	6
The brown fox	FOX	10
The brown fox	bRO	4
T	bRO	-1
The bras basah complex	bRO	-1

5. Write a function (static method) that would take in an integer and return the hexadecimal. Print out the Hex of all numbers 1 to 100 and compare your answer with that of the built in function.

Input	Output
0	0
15	F
16	10
100	64

6. Write a static method: Substitute(string s, char c1, char c2) that would return a string. The method should find all occurrences of the character c1 in the string s and substitute c1 with character c2. The new word so formed would be the return value of this method.

7. Write a static method: SetArray(int[] arr, int value) that would assign the value into all the elements of the array arr.

8. Write a static method: ResizeArray(int[] arr, int newSize) that would return a new array with the new size and copy all the content of the old array to the new array.

9. Write a static method: IsPrime(int n) that would return a boolean and use IsPrime method to print out prime number from 5 to 1000

10. Write a static method MatrixMultiply(int[,] A, int [,] B) that will perform a matrix multiplication and return another 2 dimensional array. Matrix multiplication is done as follows:

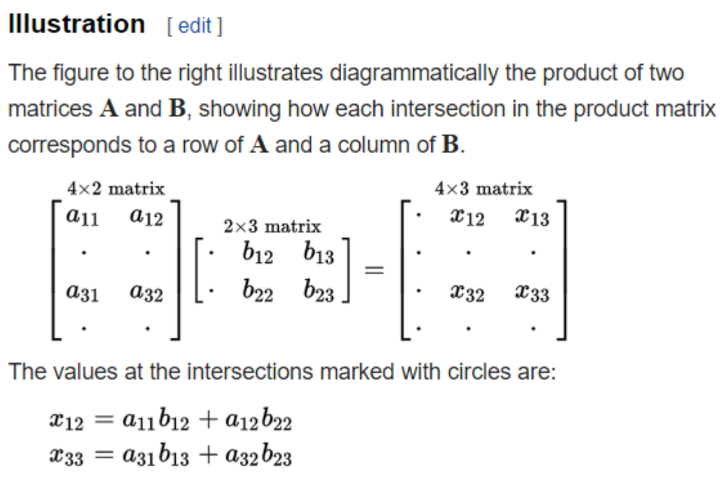


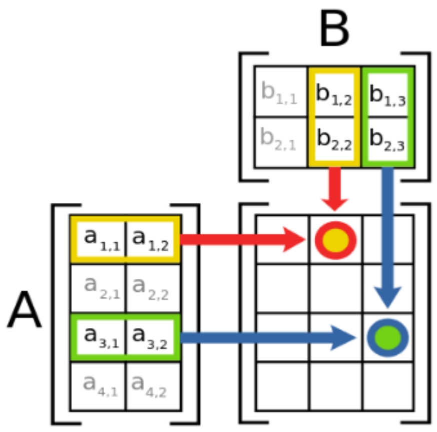
Illustration [edit]

The figure to the right illustrates diagrammatically the product of two matrices $\mathbf{A}$ and $\mathbf{B}$, showing how each intersection in the product matrix corresponds to a row of $\mathbf{A}$ and a column of $\mathbf{B}$.

$$\overset{4\times 2 \text{ matrix}}{\begin{bmatrix} a_{11} & a_{12} \\ \cdot & \cdot \\ a_{31} & a_{32} \\ \cdot & \cdot \end{bmatrix}} \overset{2\times 3 \text{ matrix}}{\begin{bmatrix} \cdot & b_{12} & b_{13} \\ \cdot & b_{22} & b_{23} \end{bmatrix}} = \overset{4\times 3 \text{ matrix}}{\begin{bmatrix} \cdot & x_{12} & x_{13} \\ \cdot & \cdot & \cdot \\ \cdot & x_{32} & x_{33} \\ \cdot & \cdot & \cdot \end{bmatrix}}$$

The values at the intersections marked with circles are:

$$x_{12} = a_{11}b_{12} + a_{12}b_{22}$$
$$x_{33} = a_{31}b_{13} + a_{32}b_{23}$$



(Reference: Wikipedia)

Delegate problem (optional)

11. Declare a delegate: double DoubleOps(double x) that represent method that will perform some double operation.

Write a static method ProcessArray(double[] arr, DoubleOps ops) that will return an array that has the same size as arr, and apply the delegate on each of the element of arr and assign it to the corresponding elements in the new array.

Use ProcessArray method to get a new array that contains the square root of the elements of the original array and a new array that contains the square of the original array.